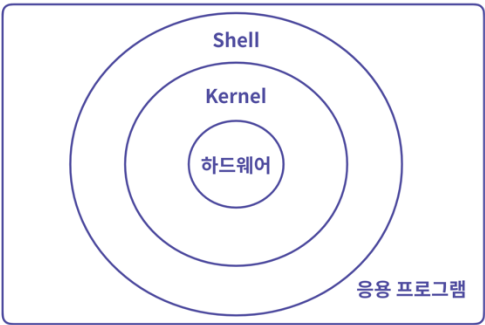


# LINUX

## 리눅스의 구조



패키지 형식		패키지 관리자	운영체제 이름
레드햇	.rpm	yum	CentOS
			페도라(Fedora)
데비안 레드햇	.deb	Apt	우분투(Ubuntu)
			리눅스 민트
			라즈비안
안드로이드	.apk	Android Package Manager	안드로이드 OS

## Shell의 종류

### Shell

Shell은 커널과 사용자 사이를 **이어주는** 역할  
사용자의 명령을 해석하여 커널에 명령을 요청

이름	키워드
Bourne-Again <b>Sh</b> ell	bash
Bourne <b>Sh</b> ell	Sh
<b>C</b> Shell	csh
<b>K</b> orn <b>Sh</b> ell	Ksh

# root

리눅스에서 모든 권한을 가지고 있는 최고 관리자

## 패키지 관리자

apt, Advanced Packaging Tool

데비안 리눅스(.dev) 또는 파생된 배포판(우분투)에서 소프트웨어를 설치, 제거, 업데이트 할 때 사용

apt-get

기존에 사용하던 키워드로 최신의 우분투 버전에서는 apt-get 과 apt-cache 가 합쳐진

apt 라는 키워드를 제공

apt install

apt remove

apt update

apt search

apt show

# 리눅스 명령어

## **whoami**

로그인한 사용자 ID 를 알려준다.

## **pwd**

현재 디렉토리 위치를 출력한다.

## **ls**

현재 디렉토리의 목록을 출력한다.

## **ls -l**

현재 디렉토리의 목록을 상세히 출력한다.

## **ls -a**

숨겨진 파일이나 디렉토리를 포함하여 출력한다.

## **ls -al**

숨겨진 파일이나 디렉토리를 포함하여 현재 디렉토리 목록을 상세히 출력한다.

## **cd**

해당 디렉토리로 이동한다.

## **--help**

사용하고자 하는 명령어 뒤에 붙여서 사용한다.

명령어에 대한 사용 방법을 출력한다.

## **sudo [사용할 명령어]**

리눅스에서 모든 권한을 가지고 있는 최고 관리자의 권한으로 명령한다.

## **tree**

하위 디렉토리를 tree 형태로 출력한다.

## **chmod [파일권한] [변경할 파일 위치 또는 이름]**

파일의 권한을 변경한다. [파일 권한]은 숫자로 표한다. 읽기 4 쓰기 2 실행 1

예) `chmod 777 elice.txt`

**chown [소유할 유저]:[소유할 그룹] [소유권을 변경하고 싶은 디렉토리 혹은 파일 이름]**  
파일의 소유권을 변경한다. root 권한이 있어야 실행할 수 있다.

**mkdir [생성할 디렉토리 이름]**  
디렉토리를 생성한다.

**touch [생성할 파일 이름]**  
빈 파일을 생성한다.

**rmdir [삭제할 디렉토리 이름]**  
디렉토리를 삭제한다. 지정한 디렉토리에 파일이 있을 경우 삭제할 수 없다.

**rm [option][삭제할 디렉토리 및 파일 이름]**  
파일 및 디렉토리 삭제를 삭제한다.  
-r 디렉토리와 그 하부 파일까지 삭제한다.  
-f 삭제 여부를 묻지 않고 바로 삭제한다.  
-i 삭제할 것인지 물어본다.  
-rf 삭제 여부를 묻지 않고 하부 파일이 있는 디렉토리까지 삭제한다.

**cp [option] [대상 위치] [복사하고 싶은 위치]**  
파일 혹은 디렉토리를 복사한다.  
-r 하위 디렉토리와 파일 전체를 복사한다.  
-p 소유주, 그룹, 권한, 시간 정보를 그대로 복사한다.

**mv [대상 위치] [복사하고 싶은 위치]**  
파일 및 디렉토리 이동한다.

**cat [option] [파일 이름]**  
파일의 내용을 출력한다.  
-n 왼쪽에 줄 번호와 함께 내용을 출력한다.  
> 파일의 내용을 덮어 쓴다.  
>> 파일의 내용이 있다면 뒤에 내용을 추가한다.

## find [파일 경로] [-name] [파일 이름] [-type d/f]

지정된 [파일 경로]에서 이름이 [파일 이름]인 파일 및 디렉토리를 검색한다.

-type d 디렉토리만 검색한다

-type f 파일만 검색한다.

## grep [option] [pattern] [파일 이름]

파일 안에서 지정한 패턴이나 문자열을 찾은 후에 그 패턴을 포함하고 있는 모든 행을 출력한다.

-i 대소문자를 구분하지 않고 검색한다.

-v 패턴과 일치하지 않는 행을 출력한다.

-c 패턴과 일치하는 행의 개수를 출력한다.

-w 패턴과 단어 단위로 매칭되어야 출력한다.

### grep 예시

```
grep elice hello.txt
```

hello.txt 파일에서 'elice'라는 문자열이 들어 있는 행을 모두 출력

```
grep -c elice hello.txt
```

hello.txt 파일에서 'elice'라는 문자열이 들어 있는 행의 수 출력

### grep과 정규 표현식

```
grep [0-9] hello.txt
```

hello.txt 파일에서 숫자가 존재하는 행을 모두 출력

```
grep "Elice\.\The\.\Rabbit" *
```

현재 디렉토리의 모든 파일에서 Elice.The.Rabbit을 찾아 출력

### grep과 정규 표현식

```
grep -v ^# somecode
```

somecode 파일에서 #으로 시작하지 않는 행을 모두 출력

```
grep '\<[a-z].*e\>' elice
```

elice 파일에서 소문자로 시작하고 공백을 포함한 여러 문자가 나오며, e로 끝나는 단어가 포함된 행을 모두 출력

**mount [option] [device] [directory]**

물리적인 저장장치를 디렉토리에 연결시켜주는 명령어. 리눅스는 PnP (Plug and Play) 기본적으로 기능이 작동하지 않기 때문에 직접 연결해야 한다.

-a /etc/fstab 에 명시된 파일 시스템을 마운트 할 때 사용한다.

-t 파일 시스템의 유형을 지정하거나 생략할 때 /etc/fstab 파일을 참조한다.

-o 추가적인 설정을 적용할 때 사용한다. 다수의 조건을 적용할 때는 ,로 구분한다

**remount [device] [directory]**

mount 를 취소한다.

**df**

현재 mount 된 디스크 정보를 출력한다.

**head**

처음 N 줄을 출력한다.

**tail**

마지막 N 줄을 출력한다.

**more**

화면단위를 출력한다.

**alias**

지정 명령어

**diff**

파일 비교

**su**

현재 사용자 변경

**which**

절대 경로

**wc**

라인, 단어의 수

**shutdown**


시스템 종료

# 프로세스

시스템에서 메모리에 적재되어 실행되고 있는 모든 프로그램.

- 모든 프로그램은 실행될 때 하나 이상의 프로세스를 갖는다.
- 병행적으로 실행이 가능하다.
- 부모, 자식 프로세스가 있게 된다.
- 커널에 의해 관리된다.
- 모든 프로세스에서는 소유자가 있다.
- 프로세스마다 식별을 위한 ID 가 부여된다.

## 프로세스의 메모리 구성

코드 영역	코드 영역 • 프로그램 코드를 뜻함
데이터 영역	데이터 영역 • 전역 변수와 정적 변수
BSS 영역	BSS 영역 • 초기화되지 않은 변수
힙 영역	힙 영역 • 동적인 메모리 할당영역
	
스택 영역	스택 영역 • 함수 매개변수, 복귀 주소, 지역 변수와 같은 임시 자료
커널	

### ps [option]

프로세스의 목록을 출력한다.

- e 현재 실행 중인 모든 프로세스 정보 출력한다.
- f 모든 정보를 확인한다
- a 실행중인 전체 사용자의 모든 프로세스를 출력한다
- u 프로세스를 실행한 사용자와 프로세스 시작 시간 등을 출력한다.
- x 터미널 제어 없이 프로세스 현황을 보여준다.

### kill [option] [PID]

-l 사용 가능한 시그널 목록을 출력한다.

-1 재실행(SIGHUP)

-2 강제 종료(SIGKILL)

-15 정상 종료(SIGTERM)

## job

백그라운드에서 실행되는 작업을 출력한다.

- 프로세스와 달리 터미널 명령을 통한 작업만을 의미한다.
- job 을 통해 프로세스를 실행할 수 있지만 터미널이 종료되면 job 과 함께 프로세스도 종료된다.
- 각각의 터미널 마다 job 은 따로 존재한다.
- 명령어 뒤에 &을 붙이면 백그라운드에서 작업을 실행한다.

**fg [jobs id] ## 포그라운드로 프로세스 실행**

**bg [jobs id] ## 백그라운드로 프로세스 실행**

## job



job을 활용하여 프로세스를 더 효율적으로 관리할 수 있다.



## job

```
kill %작업번호
```

프로세스와 마찬가지로 kill 명령어를 통해 종료 가능

ps 명령어를 통해 PID를 알아내어 종료하는 방법도 가능

## job

```
elice@elice-VirtualBox:~$ sleep 500 &
[1] 2292
elice@elice-VirtualBox:~$ sleep 700 &
[2] 2293
elice@elice-VirtualBox:~$ jobs
[1]-  실행중           sleep 500 &
[2]+  실행중           sleep 700 &
elice@elice-VirtualBox:~$ kill %1
elice@elice-VirtualBox:~$ jobs
[1]-  종료됨           sleep 500
[2]+  실행중           sleep 700 &
elice@elice-VirtualBox:~$
```

## at [option] [시간] [날짜] [+증가 시간]

지정된 시간에 1 회 실행되는 작업 예약 명령어. 시간이 되면 수행되고 작업 리스트에서 사라진다.

-m 출력 결과가 없더라도 작업이 완료될 때 사용자에게 메일을 보낸다.

-f 스크립트 파일 등을 실행할 때 사용한다.

-l 예약된 작업의 목록을 출력한다, atq 명령어 또한 같은 동작을 수행한다

-v 작업이 수행될 시간을 출력한다.

-d 예약된 작업을 삭제한다. atrm 명령어 또한 같은 동작을 수행한다.

## at

```
elicer@a2e7c1ee376:~$ at now + 3 hours -f elice.sh
```

지금으로부터 3시간 후에 elice.sh 스크립트를 실행

## atq

```
elicer@a2e7calee376:~$ atq
4    Sat Oct 19 15:07:00 2019 a elice
3    Fri Sep 20 17:06:00 2019 a elice
2    Mon Jun 12 12:00:00 2019 a elice
```

## atrm

```
elicer@a2e7calee376:~$ atrm 4
elicer@a2e7calee376:~$ atq
3    Fri Sep 20 17:06:00 2019 a elice
2    Mon Jun 12 12:00:00 2019 a elice
```

### crontab [option]

지정된 시간에 1 회 실행되는 at 과 달리 지정된 시간에 따라 주기적으로 실행한다.

-l 현재 계정의 설정된 crontab 정보를 보여준다.

-e 현재 계정의 crontab 정보를 수정한다.

-r 현재 계정의 crontab 정보를 모두 삭제한다.

-u 특정 사용자의 crontab 정보를 다루게 해준다. root 권한이 필요하다.

## crontab 등록 형식

분(0-59) 시(0-23) 월(1-12) 요일(0-6) 수행할 명령어

요일의 경우

0: 일요일, 1: 월요일, ..., 6: 토요일의 형식으로 사용

## crontab

```
elicer@a2e7ca1ee376:~$ crontab -l
# 매월 1일 1시 10분에 /home/elice/test.sh 실행
10 1 1 * * /home/elice/test.sh
# 월요일 1시, 1시 30분에 /home/elice/backup.sh 실행
0, 30 1 * * 0 /home/elice/backup.sh
```

coma(,)를 통해 2가지 이상의 시간 표현 가능

# Standard Stream

일반적으로 표준 입력, 표준 출력, 표준 오류 출력으로 분류  
스트림은 문자열로 콘솔에 출력되도록 설정되어 있음

**stdin**

키보드 입력

**stdout**

화면 출력

**stderr**

오류 내용 출력

## File Redirection

표준 스트림(Standard Stream)의 흐름을 바꾸어 일반적인 표준 입력 및 출력 그리고 오류를 사용하지 않고, 다른 경로인 파일로 다시 지정하는 것을 말한다.

### File Redirection

```
ls > ls.txt
```

> 연산자는 표준 출력을 재지정

```
ls >> ls.txt
```

>> 연산자는 파일이 존재하지 않다면 파일을 생성,  
존재한다면 파일내용을 지우지 않고 이어서 작성

### File Redirection

표준 오류는 연산자를 사용하지 않으며

파일디스크립터 번호를 > 앞에 작성해서 사용

0 표준 입력

1 표준 출력

2 표준 에러

### File Redirection

표준 오류는 연산자를 사용하지 않으며

파일디스크립터 번호를 > 앞에 작성해서 사용

0 표준 입력

1 표준 출력

2 표준 에러

### File Redirection

```
elice@elice-VirtualBox: ~  
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)  
elice@elice-VirtualBox:~$ cd helloelice > err.txt  
bash: cd: helloelice: 그런 파일이나 디렉터리가 없습니다  
elice@elice-VirtualBox:~$ cat err.txt  
elice@elice-VirtualBox:~$ cd helloelice 2> err.txt  
elice@elice-VirtualBox:~$ cat err.txt  
bash: cd: helloelice: 그런 파일이나 디렉터리가 없습니다  
elice@elice-VirtualBox:~$
```

### File Redirection

```
cat < hello.txt > elice.txt
```

1. hello.txt의 내용을 cat 명령어의 입력 스트림으로 전송
2. cat 명령어는 hello.txt 파일의 내용을 출력
3. cat 명령어의 출력 스트림을 elice.txt로 변경
4. cat 명령어의 출력 스트림은 화면이 아닌 elice.txt에 저장

## Linux Pipe

Pipe는 둘 이상의 명령어를 묶어 출력의 결과를

다른 명령으로 전환할 수 있는 기능

명령어의 표준 출력을 또 다른 명령어의 표준 입력으로 연결

| 기호를 사용하여 명령어와 명령어를 연결

## Linux Pipe

```
head a.txt | grep [0-9]
```

1. head 명령을 실행하여 a.txt의 첫 10줄을 출력
2. 출력된 결과를 | (pipe)를 통해 grep 명령으로 전달
3. 숫자가 포함된 행을 가진 행의 결과가 모두 출력

## Linux Pipe

```
alice@alice-VirtualBox:~/문서/log$ head a.txt | grep [0-9]
3 * 3 is 9
20190903
Welcome 2 Ellice!
alice@alice-VirtualBox:~/문서/log$
```

pipe를 통해 복잡한 작업을 보다 간단히 수행 가능

## Pipe & Redirection

```
head a.txt | grep [0-9] > result.txt
```

File Redirection, Pipe를 조합하여 더욱 다양하고

효율적인 작업을 수행할 수 있다.

## Pipe & Redirection

```
ls | grep user01 > output.txt
```

1. ls 명령을 실행 (현재 디렉토리의 디렉토리/파일 명 출력)
2. ls 명령의 결과를 입력 값으로 한 grep 명령어 실행
3. user01이라는 이름을 가진 파일의 결과가 출력
4. grep 명령어의 출력 스트림은 output.txt에 저장

## Pipe & Redirection

```
alice@alice-VirtualBox:~/문서/log$ ls | grep user01 > output.txt
alice@alice-VirtualBox:~/문서/log$ cat output.txt
user02_20190807
user02_20190813
user02_20190818
user02_20190830
user02_20190910
user02_20190924
alice@alice-VirtualBox:~/문서/log$
```

output.txt 파일 안에 수행한 명령의 결과가 저장

# 나노 텍스트 에디터

## 나노 텍스트 에디터

나노 텍스트 에디터는 UNIX 호환 시스템에서 사용 가능한

가볍고 간단한 텍스트 에디터

## 나노 텍스트 에디터

```
nano [편집하려는 파일 이름]
```

다음과 같은 명령으로 나노 텍스트 에디터 실행

## 단축키

단축키	동작	단축키	동작
Ctrl + O	저장하기	Ctrl + K	저장하기
Ctrl + X	종료하기	Ctrl + U	종료하기
Ctrl + W	검색하기	Ctrl + ^	여러 줄 선택

대문자로 입력하지 않아도 된다.

# 파일

## 파일의 권한

소유자			그룹			그 외 사용자		
r	w	-	r	-	-	r	-	-
4	2	0	4	0	0	4	0	0
6			4			4		

## 파일 속성

-	rw-r-r-	1	elice	elice	8980	9월 16 09:35	elice.txt
↓	↓	↓	↓	↓	↓	↓	↓
파일 유형	파일 권한	링크 수	파일 소유자	소유 그룹	파일 크기 (Byte)	마지막 변경 시간	파일 이름



# SSH, Secure Shell

네트워크를 통해 다른 컴퓨터에 접근하거나 그 컴퓨터에서 명령을 실행하게 하는 프로토콜. SSH 를 통해 다른 컴퓨터에서 네트워크를 통해 다른 리눅스에 접속하여 명령어 및 프로그램을 실행할 수 있다.

ssh [서버 ID]@[IP Address 혹은 서버 이름 혹은 도메인]

## ssh 서버 실행하기

```
elice@elice-VirtualBox:~$ sudo service ssh start
elice@elice-VirtualBox:~$ service --status-all | grep +
[ + ] acpid
[ + ] alsa-utils
[ + ] apparmor
[ + ] apport
[ + ] atd
[ + ] avahi-daemon
[ + ] cron
[ + ] cups
[ + ] cups-browsed
[ + ] dbus
[ + ] gdm3
[ + ] grub-common
[ + ] kerneloops
[ + ] kmod
[ + ] network-manager
[ + ] networking
[ + ] procs
[ + ] rsyslog
[ + ] speech-dispatcher
[ + ] ssh
[ + ] udev
[ + ] ufw
[ + ] unattended-upgrades
[ + ] whoopie
elice@elice-VirtualBox:~$
```

]

## ssh 포트 확인하기

```
elice@elice-VirtualBox:~$ sudo netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
361/systemd-resolve
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
3846/sshd
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
3156/cupsd
tcp6       0      0 :::22                   :::*                     LISTEN
3846/sshd
tcp6       0      0 :::1:631                :::*                     LISTEN
3156/cupsd
elice@elice-VirtualBox:~$
```